Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

Software-Intensive Systems Producibility

Grady Campbell
<ghc@sei.cmu.edu>

Sponsored by the U.S. Department of Defense

Report Documentation Page		*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE **MAY 2006**	2. REPORT TYPE	3. DATES COVERED **00-00-2006 to 00-00-2006**
4. TITLE AND SUBTITLE **Software-Intensive Systems Producibility**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University ,Software Engineering Institute (SEI),Pittsburgh,PA,15213**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR’S ACRONYM(S)
		11. SPONSOR/MONITOR’S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES **Systems and Software Technology Conference 2006 (SSTC 2006), Salt Lake City, UT, 1-4 May 2006.**		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**17**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Producibility

The ability to deliver needed capability in a timely, cost-effective, and predictable manner

Dimensions of Producibility

Developer productivity (efficiency and effectiveness)

- Domain knowledge and expertise, effective methods
- Engineering discipline, process capability
- Technology base (applicability, effort reduction)
- Leveragable resources (legacy, COTS, open source)
- Addressing uncertainty, diversity, and change in needs

Product utility and quality

- Functionality responsive to mission needs
- Quality attributes as determinants of system properties
- Compatibility with system and operational environment

Acquirer acuity

- Producibility-enabling acquisition policies and practices
- Effective system-software engineering synergy
- Mechanisms for capability-cost-schedule predictability
- Infrastructure for product transition-into-use and evolution

Viewpoints on Problems/Solutions

High Confidence Software and Systems Research Needs, ITRD HCSS Coordinating Group, 2001 <http://www.nitrd.gov/subcommittee/hcss.html>

New Visions for Software Design & Productivity: Research & Applications (Vanderbilt Workshop), ITRD SDP Coordinating Group, 2001 <http://www.nitrd.gov/subcommittee/sdp.html>

Report on the 2005 Software Producibility Workshops (Draft 12/12/05), DUSD(S&T), 2005

An OSD Strategy for Producibility

- Formulate a framework that gives a rationale basis for prioritizing research in producibility
- Coordinate and jointly sponsor basic and applied research to address specific producibility challenges
- Actively transition research successes into DoD use:
  - Establish centers for coordinating and assisting the maturation, validation, and packaging of research results into deliverable technology
  - Promote Federal/DoD acquisition practices that facilitate program-level adoption of producibility advances
  - Charter DoD technology transition agents for advocating and supporting program-level adoption of technology

A Reference Vision for Producibility

CAD/CAM for software-intensive systems

Model-centric – All problem/solution information is represented in a comprehensive multi-faceted product model

Virtualized – A system is defined by building, pre-deploying, and validating it in software for a hybrid hardware/software virtual environment

Predictable – Software and dependent system properties of interest are able to be accurately predicted and mutually optimized

Decision-focused – Multiple alternative solutions can be modeled, produced, and empirically evaluated based on identified customer/engineering choices

Evolvable – The problem/solution model can be continuously evolved to produce product variants that meet anticipated changing needs

Long-term Producibility Goals

Obtain a development environment in which multiple delivery-ready versions of a product can each be built and verified within 3 months

Create a capability to isolate unprecedented needs and formulate and explore alternative solutions within fixed time and resource constraints

Specify the cost and capabilities of proposed software in domain-specific models that enable a customer to negotiate a best fit to perceived needs

Be able to deliver software with zero defects operating in the customer's environment, consistent with associated domain-specific models

Be able to deliver software with precisely specified quality attributes relative to different operating conditions and verified in realistic operational use

Have the means to define how provided software will be changeable and the associated degree of effort required

Near-term Producibility Opportunities

Define a standardized framework for precisely identifying and measuring critical software-system properties

Create a DoD-wide repository exhibiting large-scale use of effective software methods for requirements specification, architectural and component design, and verification

Reformulate relevant systems and software methods to foster collaborative software-based systems engineering

Identify Federal/DoD acquisition practices that motivate programs to adopt practices that address common lifecycle software problems early

Establish a DoD capability for faciliting the packaging and transition of effective R&D technology into use on acquisition programs

Initiate efforts on programs building multi-version solutions to create a model-driven development capability based on product line principles

Current Producibility Activities

Broad Area Announcements

- Systems and Software Test Track

SBIR Topics

- Design Visualization
- Malicious Code Diffuser
- Robust Complex Systems
- Software Test Engineering
- Software Hub for High Assurance Model-Driven Development and Analysis
- Software Verification

STTR Topics

- Error Handling Paths and Policy Analysis
- Security Escorts for Not-Yet-Trusted Software
- Software System Reliability Analysis
- Assessing Interoperability Through Cross-Domain Protocol Compatibility Analysis

HPEC-SI

- Signal Processing Library

5 Themes for Funding Research

- Disciplined engineering methods
  Increase engineering discipline in the interdependent development of software and systems
- Model-based development
  Bridge the conceptual gap between domain experts and product developers
- System virtualization
  Reduce the effort to pre-verify real world behavior of software and systems
- Predictable software attributes
  Build software and systems whose properties are predictable and adjustable
- Infrastructure and emerging technology
  Adapt producibility advances to exploit or accommodate changes in infrastructure and enabling technologies

Disciplined Engineering Methods

- Management: How is iterative, concurrent, multi-version development planned, monitored, and controlled to ensure meeting schedule/budget/quality goals?
- Requirements: How do developers accurately and concisely represent the capabilities and limitations of a system/software being produced?
- Architectural design: What forms are sufficient to define the structure and composition of software in a system, as a basis for achieving tradeoffs?
- Component design: What information does an implementor need to be provided in order to build, use, or safely modify a software component?
- Implementation: How can implementation practices be improved to eliminate defects and reduce rework due to requirements or design changes?
- Verification: How can inconsistencies be precluded among multiple changing software representations (code, models/specifications, documents, tests)?
- Product families: How can an envisioned set of similar/evolving products be represented to eliminate redundant development efforts?

Carnegie Mellon
Software Engineering Institute

Model-based Development

Model: A representation of a product that enables approximate answers to a designated set of questions about the product

- **Representation: what problem-solution information is needed to define a system, what purposes should it serve, and how should it be represented?**
- **Problem analysis and specification: can the problem-solution space be abstracted into domain-specific representations to reduce the development process to an iteratively converging decision process?**
- **Solution analysis and validation: what capabilities are needed to permit rapid visualization and empirical resolution of solution alternatives for a specified problem?**
- **Product generation: What mechanisms will enable rapid correct generation of customized software, documentation, and support materials from a model?**
- **Model-product verification: what capabilities are needed to ensure that a derived solution product correctly implements a model?**

System Virtualization

- Platform independence: What form should implementations take to permit alternative physical realizations while avoiding unnecessary dependence on any specific realization?
- Hardware abstraction: How is hardware represented to enable simulated use for verifying software and supporting hardware/software codesign?
- Environment simulation: How can capabilities and constraints of the operational environment (systems, devices) be represented to enable simulated use for software validation?
- Usage simulation: How are potential uses of a system adequately represented to enable realistic automated testing?
- System validation: What techniques enable validation of a solution under realistic (normal and degraded) conditions?

Predictable Software Attributes

Design => Analysis of Alternatives and Risk Mitigation

- Identification: What are critical software-affected attributes and how do they interact? (Performance, reliability, availability, security, safety, usability, …?)
- Measurement: What behavior does the system exhibit in terms of critical attributes and how is this determined by implemented software?
- Prediction: Given a proposed software design and implementation, how will critical attributes be affected?
- Optimization: What design/implementation decisions lead to the best combination of critical attribute values?

Infrastructure & Emerging Technology

How is producibility enhanced or changed due to capabilities of computing infrastructure and emerging technologies such as these?

- **Computational technology: Multi-core processors; distributed processing, services, and data; autonomous agents; grid computing**
- **Componentization: Packaged pluggable components and frameworks (COTS, legacy, open-source; defined interfaces outward & underneath)**
- **Customization: Total-product variant and configuration management; product family and generator techniques**
- **Commoditization: Standardized cross-domain frameworks for common system capabilities**
- **Cross-speciality collaborative engineering: Tools and techniques for communication and coordination**

Considerations for Investment

Payoff: Directly addresses some aspect of the producibility problem as experienced by DoD programs

Timeframe: Offers specific near-term benefit to current DoD programs

Pragmatics: Compatible with current DoD practices and emerging technology trends and not dependent on other advances that are not timely

Research opportunity: Not otherwise adequately funded relative to DoD needs

Transitionability: Credible plan for packaging into a discretely adoptable form for near-term transition

Transition opportunity: Identified DoD acquisition programs anticipate near-term benefit and agree to evaluate utility

A Notional Producibility Task Timeline

- Propose a viable task for near-term progress on a producibility issue [1 month]
- Develop and demonstrate proof-of-concept [6 months]
- Propose a follow-on task to develop, package, and deliver/support technology [2 months]
  - Define a business case and identify a DoD/industry transition agent and DoD programs targeted for adoption
  - Refine the approach and plan for development and transition-to-practice
- Develop and package transferrable technology [12 months]
- Support adoption and use of packaged technology by targeted DoD programs/industry, and improve [12 months]